Removing the Bias from Line Detection

Carsten Steger
Forschungsgruppe Bildverstehen (FG BV)
Informatik IX, Technische Universität München
Orleansstr. 34, 81667 München, Germany
E-mail: stegerc@informatik.tu-muenchen.de

Abstract

The extraction of curvilinear structures is an important low-level operation in computer vision. Most existing operators use a simple model for the line that is to be extracted, i.e., they do not take into account the surroundings of a line. Therefore, they will estimate a wrong line position whenever a line with different lateral contrast is extracted. In contrast, the algorithm proposed in this paper uses an explicit model for lines and their surroundings. By analyzing the scale-space behavior of a model line profile, it is shown how the bias that is induced by asymmetrical lines can be removed. Thus, the algorithm is able to extract an unbiased line position and width, both with sub-pixel accuracy.

1. Introduction

Extracting curvilinear structures, often simply called lines, in digital images is an important low-level operation in computer vision that has many applications. In photogrammetric and remote sensing tasks it can be used to extract linear features, including roads, railroads, or rivers, from satellite or low resolution aerial imagery, which can be used for the capture or update of data for geographic information systems [1]. In addition, it is useful in medical imaging for the extraction of anatomical features, e.g., blood vessels from an X-ray angiogram [3].

The published schemes for line detection can be classified into two categories. For a detailed review of these methods see [14]. The first approach is to regard lines as objects having parallel edges [8, 5], and to extract them by combining the output of two specially tuned edge filters. By iteration in scale-space, lines of arbitrary widths can be detected, but only with limited width resolution. Furthermore, the approach is computationally very expensive.

The second approach is to regard the image as a function $z(x, y)$ and extract lines from it by using various differential geometric properties of this function. The basic idea behind these algorithms is to locate the positions of ridges and ravines in the image function. These methods can be further divided according to which property they use.

In the first sub-category, ridges are found at points where one of the principal curvatures of the image function assumes a local maximum [10, 4]. For lines with a flat profile it has the problem that two separate points of maximum curvature symmetric to the true line position will be found [4].

In the second sub-category, ridges and ravines are detected by locally approximating the image function by its second or third order Taylor polynomial. The coefficients of this polynomial are usually determined by using the facet model, i.e., by a least squares fit of the polynomial to the image data over a window of a certain size [2, 15, 6] or by using derivatives of Gaussian masks [9, 4]. The approaches using the facet model have the problem that only lines of a certain width can be extracted [14], while the Gaussian masks can be tuned for a certain line width by selecting an appropriate σ. It is also possible to select the appropriate σ for each image point by iterating through scale space [9]. However, since the surroundings of the line are not modeled the extracted line position becomes progressively inaccurate as σ increases.

Very few approaches to line detection consider the task of extracting the line width along with the line position. Most of them do this by an iteration through scale-space while selecting the scale, i.e., the σ, that yields the maximum value to a certain scale-normalized response as the line width [8, 9]. However, this is computationally very expensive, especially if one is only interested in lines in a certain range of widths. Furthermore, these approaches will only yield a relatively rough estimate of the line width since, by necessity, the scale-space is quantized in rather coarse intervals. Finally, the scale-space analysis presented in this paper shows that the position of the extracted line will generally be biased for the optimum scale σ. A different approach is given in [2], where lines and edges are extracted in one simultaneous operation. For each line point two corresponding edge points are matched from the resulting description. This approach has the advantage that lines and their corresponding edges can in principle be extracted with sub-pixel accuracy. However, since the approach does not use an explicit model for a line, the location of the corresponding edge of a line is often not meaningful because the interaction between a line and its corresponding edges is neglected.

2. Detection of Line Points

2.1. Models for Line Profiles in 1D

Many approaches to line detection consider lines in 1D to be bar-shaped, i.e., the ideal line of width $2w$ and height h is assumed to have a profile given by

$$f_b(x) = \left\{ \begin{array}{ll} h, & |x| \le w \\ 0, & |x| > w \ . \end{array} \right. \tag{1}$$

However, the assumption that lines have the same contrast on both sides is rarely true for real images. Therefore, asymmetrical bar-shaped lines

$$f_a(x) = \left\{ \begin{array}{ll} 0, & x < -w \\ 1, & |x| \le w \\ a, & x > w \ . \end{array} \right. \tag{2}$$

($a \in [0,1]$) are considered as the most common line profile in this paper. General lines of height h can be obtained by considering a scaled asymmetrical profile, i.e., $hf_a(x)$.

2.2. Detection of Line Profiles in 1D

The approach to extract individual line points has been described in [12, 14]. Therefore, only a brief summary of the approach as far as they are relevant to the rest of the paper are given here. In order to detect lines with a profile given by (1) or (2) the image should be convolved with the derivatives of the Gaussian kernel. For the symmetrical bar-shaped profile (1) this leads to the following description of the behavior of the derivatives in scale-space:

$$\begin{aligned} r_b(x,\sigma,w,h) &= h(\phi_\sigma(x+w) - \phi_\sigma(x-w)) & (3) \\ r_b'(x,\sigma,w,h) &= h(g_\sigma(x+w) - g_\sigma(x-w)) & (4) \\ r_b''(x,\sigma,w,h) &= h(g_\sigma'(x+w) - g_\sigma'(x-w)) \ , & (5) \end{aligned}$$

where $g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2})$ is the Gaussian with standard deviation σ, and $\phi_\sigma(x)$ is its integral.

In order to detect line points it is sufficient to determine the points where $r_b'(x,\sigma,w,h)$ vanishes. However, in order to select only salient lines, the magnitude of the second derivative $r_b''(x,\sigma,w,h)$ in the point where $r_b'(x,\sigma,w,h) = 0$ should be significantly different from 0. When the scale-space behavior of $r_b''(x,\sigma,w,h)$ is analyzed it can be seen that

$$\sigma \ge w/\sqrt{3} \tag{6}$$

has to be observed in order for the line point extraction to be reliable for lines of a width smaller than $2w$ [12, 14].

From (1) and (4) it is also evident that a line is bounded by an edge on each side of the line. Hence, to detect the line width the edge points to the left and right of the line point need to be extracted [13, 14]. Their position is given by the solutions of $r_b''(x,\sigma,w,h) = 0$, where $r_b'''(x,\sigma,w,h) < 0$, i.e., the points that exhibit a maximum in the absolute value of the gradient.

2.3. Detection of Lines in 2D

The method to extract individual line points in 2D has been detailed in [12, 14]. Again, only a brief summary is given. Curvilinear structures in 2D can be modeled as curves $s(t)$ that exhibit the characteristic 1D line profile f_a in the direction perpendicular to the line, i.e., perpendicular to $s'(t)$. Let this direction be $n(t)$. This means that the first directional derivative in the direction $n(t)$ should vanish and the second directional derivative should be of large absolute value. The direction $n(t)$ can be determined by calculating the eigenvalues and eigenvectors of the Hessian matrix $H(x,y)$ of the image after convolution with the appropriate derivatives of the 2D Gaussian kernel. The extraction can be done with sub-pixel accuracy.

To detect the width of the line, for each line point the closest points in the image to the left and to the right of the line point, i.e., along $-n$ and n, where the absolute value of the gradient takes on its maximum value need to be determined. Equation (6) shows that it is sensible to search for edges only in a restricted neighborhood of the line. Ideally, the maximum distance for the search would be $\sqrt{3}\sigma$. In order to ensure that most of the edge points are detected, the current implementation uses a slightly larger maximum distance of 2.5σ [13, 14].

An important issue is what the algorithm should do when it is unable to locate an edge point for a given line point. This might happen, for example, if there is a very weak and wide gradient next to the line, which does not exhibit a well defined maximum. Another case where this typically happens are the junction areas of lines, where the line width usually grows beyond the range of 2.5σ. A good solution to this problem is to interpolate or extrapolate the line width from neighboring line points [14].

2.4. Examples

Figure 1(b) displays the results of the line extraction algorithm for the image of Fig. 1(a). This image is fairly good-natured in the sense that the lines it contains are rather symmetrical. From Fig. 1(a) it can be seen that the algorithm is able to locate the edges of the wider line with very

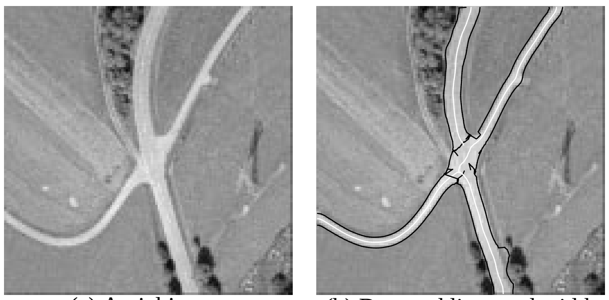

(a) Aerial image (b) Detected lines and width

Figure 1. Lines and their width detected (b) in an aerial image (a). Lines are displayed in white while the corresponding edges are displayed in black.

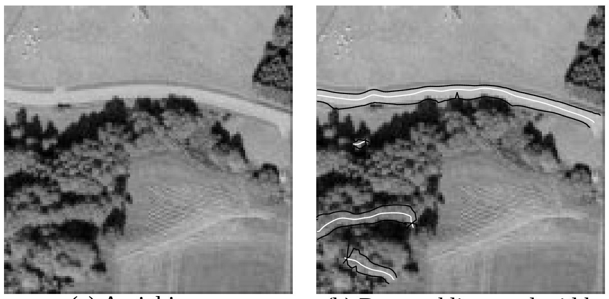

(a) Aerial image (b) Detected lines and width

Figure 2. Lines and their width detected (b) in an aerial image (a).

high precision. The only place where the edges do not correspond to the semantic edges of the road object are in the bottom part of the image, where nearby vegetation causes a strong gradient and causes the algorithm to estimate the line width too large. Please note that the width of the narrower line is extracted slightly too large. How to remove this effect is the topic of Section 3. A final thing to note is that the algorithm extrapolates the line width in the junction area in the middle of the image, as discussed in Section 2.3. This explains the seemingly unjustified edge points in this area.

Figure 2(b) exhibits the results of the proposed approach on another aerial image of the same ground resolution (1 m), given in Fig. 2(a). Please note that the line in the upper part of the image contains a very asymmetrical part in the center part of the line due to shadows of nearby objects. Therefore, the line position is shifted towards the edge of the line that posesses the weaker gradient, i.e., the upper edge in this case. Please note also that the line and edge positions are very accurate in the rest of the image.

The next Section will explore the cause for the shift in line position and for the above mentioned problem of extracting a too large line width.

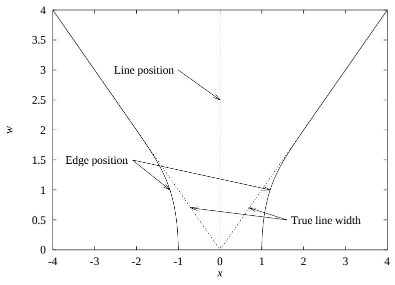


Figure 3. Location of a line with width $w \in [0,4]$ and its edges for $\sigma = 1$.

3. Removing the Bias from Asymmetric Lines

3.1. Reasons for Bias in Line Position and Width

Let us first consider the reason why the line width of narrow lines will be estimated too large for symmetrical bar-shaped profiles. Equations (4) and (5) can be used to derive how the edges of a line will behave in scale-space. Since this analysis involves equations which cannot be solved analytically, the calculations must be done using a root finding algorithm [11]. Figure 3 shows the location of the line and its corresponding edges for $w \in [0,4]$ and $\sigma = 1$. Note that the ideal edge positions are given by $x = \pm w$. From (5) it is apparent that the edges of a line can never move closer than σ to the real line, and thus the width of the line will be estimated significantly too large for narrow lines.

By far the greater problem is the bias in the extracted line position for asymmetrical lines given by (2). The responses to this kind of profile are given by:

$$\begin{aligned} r_a(x,\sigma,w,a) &= \phi_\sigma(x+w) + (a-1)\phi_\sigma(x-w) \quad (7)\\ r'_a(x,\sigma,w,a) &= g_\sigma(x+w) + (a-1)g_\sigma(x-w) \quad (8)\\ r''_a(x,\sigma,w,a) &= g'_\sigma(x+w) + (a-1)g'_\sigma(x-w) . \quad (9)\end{aligned}$$

The location where $r'_a(x,\sigma,w,a) = 0$, i.e., the position of the line, is

$$l = -\frac{\sigma^2}{2w}\ln(1-a) \quad . \qquad (10)$$

This means that the line will be estimated in a wrong position whenever the contrast is significantly different on both sides of the line. The estimated position of the line will be within the actual boundaries of the line as long as

$$a \le 1 - e^{-\frac{2w^2}{\sigma^2}} \quad . \qquad (11)$$

The location of the corresponding edges can again only be computed numerically. Figure 4 gives an example of the line and edge positions for $w = 1$, $\sigma = 1$, and $a \in [0,1]$. It can be seen that the position of the line and the edges is greatly influenced by line asymmetry. As a gets larger the

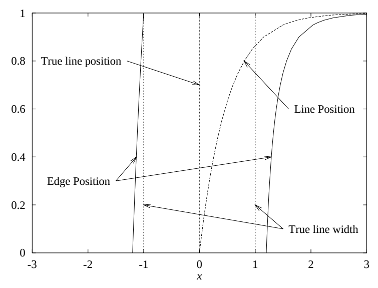


Figure 4. Location of an asymmetrical line and its corresponding edges with width $w = 1$, $\sigma = 1$, and $a \in [0, 1]$.

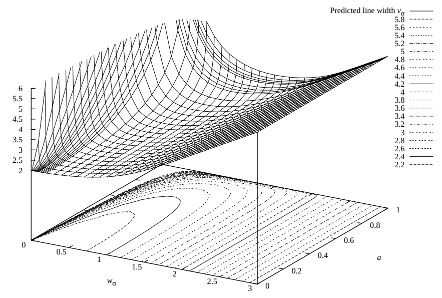
line and edge positions are pushed to the weak side, i.e., the side that posseses the smaller edge gradient.

Note that (10) gives an explicit formula for the bias of the line extractor. Suppose that we knew w and a for each line point. Then it would be possible to remove the bias from the line detection algorithm by shifting the line back into its proper position.

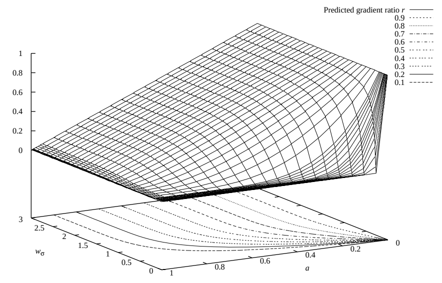
It is apparent from this analysis that failure to model the surroundings of a line, i.e., the asymmetry of its edges, can result in large errors of the estimated line position and width. Algorithms that fail to take this into account will not return meaningful results.

3.2. Detailed Analysis of Asymmetrical Line Profiles

Recall from the discussion above that if the algorithm knew the true values of w and a it could remove the bias in the estimation of the line position and width. Equations (7)–(9) give an explict scale-space description of the asymmetrical line profile f_a. The position l of the line can be determined analytically by the zero-crossings of $r'_a(x, \sigma, w, a)$ and is given in (10). The total width of the line, as measured from the left to right edge, is given by the zero-crossings of $r''_a(x, \sigma, w, a)$. Unfortunately, these positions can only be computed by a root finding algorithm since the equations cannot be solved analytically. Let us call these positions e_l and e_r. Then the width to the left and right of the line is given by $v_l = l - e_l$ and $v_r = e_r - l$. The total width of the line is $v = v_l + v_r$. The values of l, e_l, and e_r form a scale-invariant system. This means that if both σ and w are scaled by the same constant factor c the line and edge locations will be given by cl, ce_l, and ce_r (for a proof see [14]). Hence, w and σ are not independent of each other. In fact, we only need to consider all w for one particular σ, e.g., $\sigma = 1$. Therefore, for the following analysis we only need to discuss values that are normalized with regard to the scale σ, i.e., $w_\sigma = w/\sigma$, $v_\sigma = v/\sigma$, and so on. A useful consequence is that the behavior of f_a can be analyzed



(a) Predicted line width



(b) Predicted gradient ratio

Figure 5. Predicted behavior of the asymmetrical line f_a for $w_\sigma \in [0, 3]$ and $a \in [0, 1]$. (a) Predicted line width v_σ. (b) Predicted gradient ratio r.

for $\sigma = 1$. All other values can be obtained by a simple multiplication by the actual scale σ.

With all this being established, the predicted total line width v_σ can be calculated for all w_σ and $a \in [0, 1]$. Figure 5 displays the predicted v_σ for $w_\sigma \in [0, 3]$. Obviously, v_σ grows without bounds for $w_\sigma \downarrow 0$ or $a \uparrow 1$. Furthermore, it can be proved that $v_\sigma \in [2, \infty]$. Therefore, in Fig. 5 the contour lines for $v_\sigma \in [2, 6]$ are also displayed.

Section 2.3 gave a procedure to extract the quantity v_σ from the image. This is half of the information required to get to the true values of w and a. However, an additional quantity is needed to estimate a. Since the true height h of the line profile hf_a is unknown this quantity needs to be independent of h. One such quantity is the ratio of the gradient magnitude at e_r and e_l, i.e., the weak and strong side. This quantity is given by $r = |r'_a(e_r, \sigma, w, a)|/|r'_a(e_l, \sigma, w, a)|$. It is obvious that the influence of h cancels out. Furthermore, it is easy to see that r also remains constant under simultaneous scalings of σ and w. The quantity r has the advantage that it is easy to extract from the image. Figure 5 displays the predicted r for $w_\sigma \in [0, 3]$. It is obvious that $r \in [0, 1]$. Therefore, the contour lines for r in this range are displayed in Figure 5 as well. It can be seen that for large w_σ, r is very close to $1 - a$. For small w_σ it will drop to near-zero for all a.

3.3. Inversion of the Bias Function

The discussion above can be summarized as follows: The true values of w_σ and a are mapped to the quantities v_σ and r, which are observable from the image. More formally, there is a function $f : (w_\sigma, a) \in [0, \infty] \times [0, 1] \mapsto (v_\sigma, r) \in [2, \infty] \times [0, 1]$. From the discussion in Section 2.3 it follows that it is only useful to consider $v_\sigma \in [0, 5]$. However, for very small σ it is possible that an edge point will be found within a pixel in which the center of the pixel is less than 2.5σ from the line point, but the edge point is farther away than this. Therefore, $v_\sigma \in [0, 6]$ is a good restriction for v_σ. Since the algorithm needs to determine the true values (w_σ, a) from the observed (v_σ, r), the inverse f^{-1} of the map f has to be determined. Fortunately, f is invertible for all v_σ and r [14].

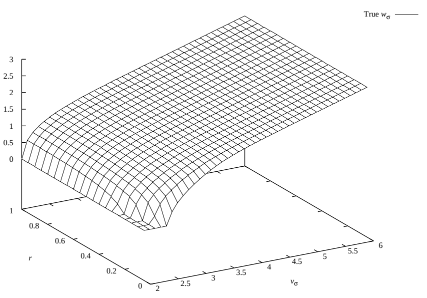
To calculate f^{-1}, a multi-dimensional root finding algorithm has to be used [11]. To obtain maximum precision for w_σ and a, this root finding algorithm would have to be called at each line point. This is undesirable for two reasons. Firstly, it is a computationally expensive operation. More importantly, however, due to the nature of the function f, very good starting values are required for the algorithm to converge, especially for small v_σ. Therefore, the inverse f^{-1} is computed for selected values of v_σ and r and the true values are obtained by interpolation. The step size of v_σ was chosen as 0.1, while r was sampled at 0.05 intervals. Figure 6 shows the true values of w_σ and a for any given v_σ and r. It can be seen that despite the fact that f is very ill-behaved for small w_σ, f^{-1} is quite well-behaved. This behavior leads to the conclusion that linear interpolation can be used to obtain good values for w_σ and a.

One final important detail is how the algorithm should handle line points where $v_\sigma < 2$, i.e., where f^{-1} is undefined. This can happen, for example, because there are two lines very close to each other. In this case the edge points cannot move as far outward as the model predicts. If this happens, the line point will have an undefined width. These cases can be handled by interpolation, similar to the case of missing edge points in Section 2.3.

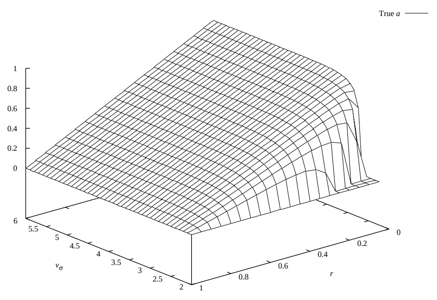
With all this information calculated it is now a simple matter to calculate the true contrast h of the line. It is given by the ratio of predicted response r''_a according to (9) and the observed response r'' in the image:

$$h = \frac{r''}{r''_a(l, \sigma, w, a)} \quad , \tag{12}$$

where l is the line position according to (10). In order to achieve maximum accuracy, r'' has to be determined with sub-pixel precision from the image by interpolation.



(a) True w_σ



(b) True a

Figure 6. True values of the line width w_σ (a) and the asymmetry a (b).

3.4. Examples

Figure 7 shows how the bias removal algorithm is able to succesfully adjust the line widths in the aerial image of Fig. 1. Please note from Fig. 7(a) that because the lines in this image are fairly symmetrical, the line positions have been adjusted only minimally. Furthermore, it can be seen that the line widths correspond much better to the true line widths. Figure 7(b) shows a four times enlarged part of the results superimposed onto the image in its original ground resolution of 0.25 m, i.e., four times the resolution in which the line extraction was carried out. For most of the lines the edges are well within one pixel of the edge in the larger resolution. Figure 7(c) shows the same detail without the removal of the bias. In this case, the extracted edges are about 2–4 pixels from their true locations. The bottom part of Fig. 7(a) shows that sometimes the bias removal can make the location of one edge worse in favor of improving the location of the other edge. However, the position of the line is affected only slightly.

Figure 8 shows the results of removing the bias from the test image of Fig. 2. Please note that in the areas of the image where the line is highly asymmetrical the line and edge locations are much improved. In fact, for a very large part of the road the line position is within one pixel of the road markings in the center of the road in the high resolution image. Again, a four times enlarged detail is shown in Fig. 8(b). If this is compared to the detail in Fig. 8(c)

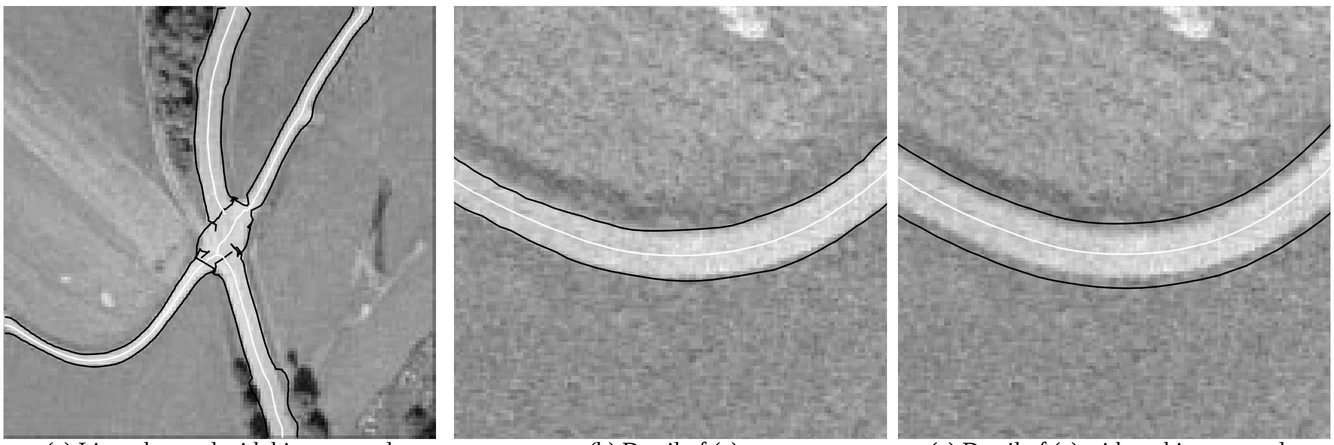

(a) Lines detected with bias removal (b) Detail of (a) (c) Detail of (a) without bias removal

Figure 7. Lines and their width detected (a) in an aerial image of resolution 1 m with the bias removed. A four times enlarged detail (b) superimposed onto the original image of resolution 0.25 m. (c) Comparison to the line extraction without bias removal.

the significant improvement in the line and edge locations becomes apparent.

4. Conclusions

This paper has presented an approach to extract lines and their widths with very high precision. A model for the most common type of lines, the asymmetrical bar-shaped line, was proposed. A scale-space analysis was carried out for this model profile. This analysis shows that there is a strong interaction between a line and its two corresponding edges which cannot be ignored. The true line width influences the line width occuring in an image, while asymmetry influences both the line width and its position. From this analysis an algorithm to extract the line position and its width was derived. This algorithm exhibits the bias that is predicted by the model for the asymmetrical line. Therefore, a method to remove this bias was proposed. The resulting algorithm works very well for a range of images containing lines of different widths and asymmetries, as was demonstrated by a number of test images. High resolution versions of the test images were used to check the validity of the obtained results. They show that the proposed approach is able to extract lines with very high precision from low resolution images. The extracted line positions and edges correspond to semantically meaningful entities in the image, e.g., road center lines and roadsides or blood vessels [14]. The approach only uses the first and second directional derivatives of an image for the extraction of the line points. No specialized directional filters are needed. The edge point extraction is done by a localized search around the line points already found using five very small masks. This makes the approach computationally very efficient. For example, the time to process an image of size 256×256 is about 1.7 seconds on a HP 735 workstation.

The presented approach shows two fundamental limitations. First of all, it can only be used to detect lines with a certain range of widths, i.e., between 0 and 2.5σ. This is a problem if the width of the important lines varies greatly in the image. However, since the bias is removed by the algorithm, one can in principle select σ large enough to cover all desired line widths and the algorithm will still yield valid results. This will work if the narrow lines are relatively salient. Otherwise they will be smoothed away in scale-space. Of course, once σ is selected so large that neighboring lines will start to influence each other the line model will fail and the results will deteriorate. Hence, in reality there is a limited range in which σ can be chosen to yield good results. In most applications this is not a very significant restriction since one is usually only interested in lines in a certain range of widths. Furthermore, the algorithm could be iterated through scale-space to extract lines of very different widths. The second problem is that the definition of salient lines is done via the second directional derivatives. However, one can use semantically meaningful values, i.e., the width and height of the line, to obtain the desired thresholds [12, 14]. Therefore, the algorithm could be modified to accept these parameters as thresholds, and to compute appropriate thresholds for the second derivative from them internally.

Finally, it should be stressed that the lines extracted are not ridges in the topographic sense, i.e., they do not define the way water runs downhill or accumulates [7]. In fact, they are much more than a ridge in the sense that a ridge can be regarded in isolation, while a line needs to model its surroundings. If a ridge detection algorithm is used to ex-

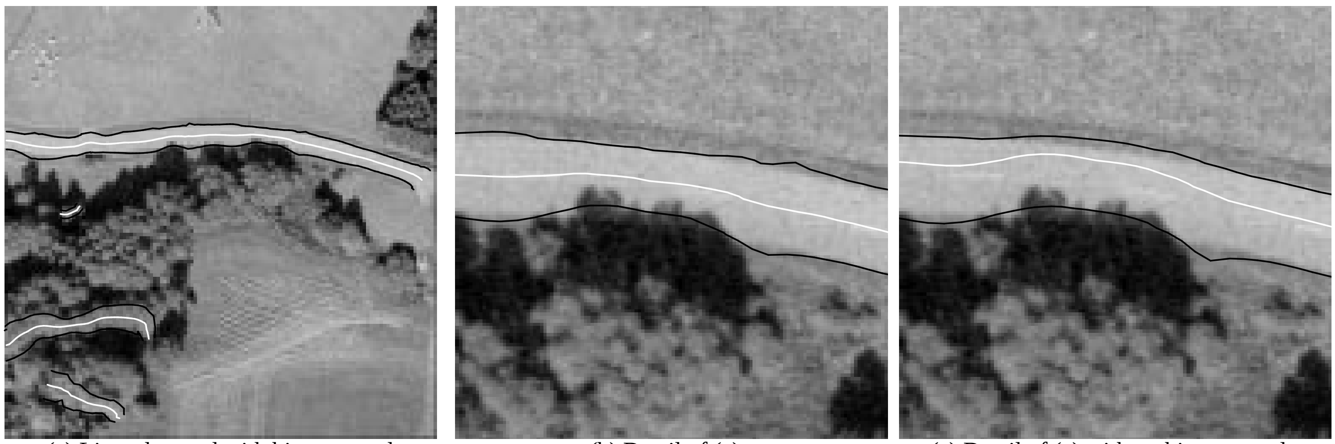

(a) Lines detected with bias removal (b) Detail of (a) (c) Detail of (a) without bias removal

Figure 8. Lines and their width detected (a) in an aerial image of resolution 1 m with the bias removed. A four times enlarged detail (b) superimposed onto the original image of resolution 0.25 m. (c) Comparison to the line extraction without bias removal.

tract lines, the asymmetry of the lines will invariably cause it to return biased results.

Further work will concentrate on applying the technique for modeling the bias in line position and width developed in this paper to the case of lines with contrast of different polarity, i.e., lines where the background is darker than the line on one side of the line and brighter on the other side. For this type of line qualitatively similar effects occur.

References

[1] A. Baumgartner, C. Steger, C. Wiedemann, H. Mayer, W. Eckstein, and H. Ebner. Update of roads in GIS from aerial imagery: Verification and multi-resolution extraction. In *International Archives of Photogrammetry and Remote Sensing*, volume XXXI, part B3, pages 53–58, 1996.

[2] A. Busch. A common framework for the extraction of lines and edges. In *International Archives of Photogrammetry and Remote Sensing*, volume XXXI, part B3, pages 88–93, 1996.

[3] C. Coppini, M. Demi, R. Poli, and G. Valli. An artificial vision system for X-ray images of human coronary trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):156–162, Feb. 1993.

[4] D. Eberly, R. Gardner, B. Morse, S. Pizer, and C. Scharlach. Ridges for image analysis. Technical Report TR93–055, Department of Computer Science, University of North Carolina, Chapel Hill, NC, USA, 1993.

[5] A. Filbois and D. Gemmerlé. From step edge to line edge: Combining geometric and photometric information. In *MVA '94 IAPR Workshop on Machine Vision Applications*, pages 87–90, 1994.

[6] R. M. Haralick, L. T. Watson, and T. J. Laffey. The topographic primal sketch. *International Journal of Robotics Research*, 2(1):50–72, 1983.

[7] J. J. Koenderink and A. J. van Doorn. Two-plus-one-dimensional differential geometry. *Pattern Recognition Letters*, 15(5):439–443, May 1994.

[8] T. M. Koller, G. Gerig, G. Székely, and D. Dettwiler. Multiscale detection of curvilinear structures in 2-d and 3-d image data. In *Fifth International Conference on Computer Vision*, pages 864–869. IEEE Computer Society Press, 1995.

[9] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. In *Computer Vision and Pattern Recognition*, pages 465–470. IEEE Computer Society Press, 1996.

[10] O. Monga, N. Armande, and P. Montesinos. Thin nets and crest lines: Application to satellite data and medical images. Rapport de Recherche 2480, INRIA, Rocquencourt, France, Feb. 1995.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, 2nd edition, 1992.

[12] C. Steger. Extracting curvilinear structures: A differential geometric approach. In B. Buxton and R. Cipolla, editors, *Fourth European Conference on Computer Vision*, volume 1064 of *Lecture Notes in Computer Science*, pages 630–641. Springer-Verlag, 1996.

[13] C. Steger. Extraction of curved lines from images. In *13th International Conference on Pattern Recognition*, volume II, pages 251–255. IEEE Computer Society Press, 1996.

[14] C. Steger. An unbiased detector of curvilinear structures. Technical Report FGBV–96–03, Forschungsgruppe Bildverstehen (FG BV), Informatik IX, Technische Universität München, July 1996.

[15] L. Wang and T. Pavlidis. Direct gray-scale extraction of features for character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1053–1067, Oct. 1993.